Оглавление

[Общее 2](#_Toc163961341)

[Структурные диаграммы 3](#_Toc163961342)

[Диаграмма классов 3](#_Toc163961343)

[Межклассовые отношения 3](#_Toc163961344)

[Диаграмма компонентов 5](#_Toc163961345)

[Диаграмма развертывания 6](#_Toc163961346)

[Диаграмма объектов 7](#_Toc163961347)

[Диаграмма пакетов 8](#_Toc163961348)

[Диаграмма составной структуры 9](#_Toc163961349)

[Диаграмма профилей 10](#_Toc163961350)

[Диаграммы поведения 11](#_Toc163961351)

[Диаграмма прецедентов 11](#_Toc163961352)

[Диаграмма деятельности 12](#_Toc163961353)

[Диаграмма состояний 13](#_Toc163961354)

[Диаграмма взаимодействий 14](#_Toc163961355)

[Диаграмма последовательности 14](#_Toc163961356)

[Диаграмма коммуникаций 14](#_Toc163961357)

[Диаграмма обзора взаимодействий 14](#_Toc163961358)

[Временная диаграмма 14](#_Toc163961359)

Общее

Диаграмма классов UML иллюстрирует структуру системы, описывая классы, их атрибуты, методы и отношения между объектами.

Типы UML диаграмм:

1. Структурные диаграммы
   1. Диаграмма классов
   2. Диаграмма составной структуры
   3. Диаграмма развертывания
   4. Диаграмма пакетов
   5. Диаграмма профилей
   6. Диаграмма объектов
   7. Диаграмма компонентов
2. Диаграммы поведения
   1. Диаграмма деятельности
   2. Диаграмма прецедентов
   3. Диаграмма состояний
   4. Диаграмма взаимодействий
      1. Диаграмма последовательности
      2. Диаграмма коммуникации
      3. Диаграмма обзора взаимодействий
      4. Временная диаграмма

Структурные диаграммы

Диаграмма классов

Описывает классы их атрибуты методы и отношения между объектами

Центральная методика моделирования. Используется во всех программно-ориентированных методах. Описывает типы объектов в системе и виды статических отношений, которые существуют между объектами.

Виды перспектив:

1. Концептуальная
   1. Диаграмма интерпретируется как описание вещей в реальном мире
   2. Считается независимой от языка
2. Спецификационная
   1. Диаграмма интерпретируется как описание абстракций
   2. Без привязки к конкретной реализации
3. Имплементационная
   1. Интерпретируется как описание реализаций на коде

Межклассовые отношения

Делится все на:

1. Ассоциация.
2. Наследование\Генерализация (такая же функция, как и в ооп)
3. Реализация (подразумевается отношение интерфейса и объектов)
4. Зависимость (Объект одного класса использует объект другого класса в своем методе.
5. Агрегация (один класс является частью другого. Подклассы не уничтожаются, когда уничтожается объединяющий класс)
6. Композиция (один класс является частью другого. Подклассы уничтожаются, когда уничтожается объединяющий класс)

Ассоциация

Ассоциация – отношение между классами. Данное меж классовое отношение позволяет соединять классы.

Кратность – количество экземпляров одного класса к другому. Бывает следующих видов:

1. Один к одному (пример: один пользователь и одна роль админа)
2. Один ко многим (пример: один пользователь отыгрывает множество ролей)
3. Многие к многим (пример: множеству юзеров доступно множество ролей)

Наследование\Генерализация

Схематическое отношение между родительским и дочерним классом. По факту является тем же самым что и в ООП.

Реализация

Реализация – подразумевает отношение интерфейса и объекта. То есть один объект реализует поведение другого. Пример:

Класс Поставщик имеет методы для покупки товара. Для того чтобы сделать реализацию, класс Клиента должен иметь метод-интерфейс, который будет запрашивать информацию у пользователя и отправлять все это в метод поставщика.

Зависимость

Зависимость – объект (объект 1) одного класса может использовать объект (объект 2) другого класса в своем методе. Объект 2 не должен быть полем класса.

Агрегация

Агрегация – вид зависимости, при которой один класс (класс 1) является частью другого класса (класс 2). То есть класс2 содержит в себе класс1. При удалении класс 2, класс1 останется

Класс1 – класс-часть

Класс2 – класс-агрегатор

Композиция

Композиция – вид агрегации, при котором послу удаления класса-агрегатора удаляются и классы-части

## Диаграмма компонентов

Показывает как компоненты соединяются вместе для формирования более крупных компонентов.

Иллюстрирует архитектуру компонентов и зависимости между ними.

## Диаграмма развертывания

Позволяет моделировать физический аспект объектной ориентированной программной системы.

Схема, которая показывает архитектуру системы как развертывания программных артефактов.

Артефакты – конкретные элементы в физическом мире (является результатом процесса разработки).

Программа моделирует конфигурацию времени выполнения в статическом представлении и визуализирует распределение артефактов в приложении.

## Диаграмма объектов

Является экземпляром диаграммы класса.

Показывает состояние системы в определённый момент времени

## Диаграмма пакетов

Показывает пакеты и зависимости между ними

## Диаграмма составной структуры

Является диаграммой компонентов. Используется при моделировании систем на микро уровне.

Изображает отдельные части вместо целых классов. Показывает внутреннюю структуру класса и возможные взаимодействия.

## Диаграмма профилей

Позволяет создавать стереотипы и определять отношение между ними

# Диаграммы поведения

## Диаграмма прецедентов

Описывает функциональные требования системы с точки зрения прецедентов. Позволяет связать то что нам нужно от системы с тем как система удовлетворяет эти потребности

## Диаграмма деятельности

Представляет рабочие процессы, поэтапных действий и действий с поддержкой выбора итераций и параллелизма. Описывает управление системой

## Диаграмма состояний

Тип диаграмм, используемых для описания поведения системы. Отображают разрешенные состояния и переходы. Позволяет визуализировать весь жизненный цикл объектов.

## Диаграмма взаимодействий

### Диаграмма последовательности

Моделирует взаимодействие объектов на основе временной последовательности. Показывает как объекты взаимодействуют с другими.

### Диаграмма коммуникаций

Используется для моделирования динамических претендентов. Показывает взаимодействие объектов.

### Диаграмма обзора взаимодействий

Фокусируется на обзоре потока управления.

### Временная диаграмма

Показывает поведение объектов в период времени